ETH zürich

Soft Robotics Laboratory

Bachelor's Thesis

Modeling and Control of Quadcopters in Ground Effect using Physics Informed Machine Learning

Aurel X. Appius

June 2022

Supervision

Prof. Dr. Robert K. Katzschmann
Mike Y. Michelis

Abstract

Quadcopters have a wide range of applications and are used for many aerial operations. Their linearized rigid body dynamics provide the basis for classical control algorithms. These controllers perform great in high altitude flight, where there are only small aerodynamical disturbances. If the quadcopter operates closer to the ground, it becomes subject to ground effect, which significantly degrades control performance. I propose a new and simple control method that replaces a part of the conventional flight controller with an feed forward compensator that accounts for ground effect disturbances. By using a variation of the Sparse Identification of Nonlinear Dynamics (SINDy) algorithm, quadcopter flight data is analyzed and an accurate model is found for my quadcopter architecture. Instead of proposing another low-performing analytical ground effect model, I provide a machine learning framework for other scientist and engineers to easily find a precise ground effect model for their quadcopters.
Using the custom control architecture and the discovered model, I was able to reduce the error caused by ground effect by 78.5 %, which is significantly higher than other models found in literature, that only achieved an error reduction of 36.9 %. Furthermore I could show that the model and controller also perform well in scenarios not present in the training data, which rules out the possibility of an over fitted model.

Contents

Glossary

Blade Element Theory	is a mathematical theory that studies the behavior of propellers.
MATLAB	is a numerical computer tool for matrix manipulation and data analysis.
MAVLink	is a very lightweight messaging protocol for communicating with drones and between onboard drone components
MAVSDK	is a collection of libraries for various programming languages to interface with MAVLink systems such as drones
Pixhawk 4	is an advanced autopilot that runs the PX4 firmware
PX4	is an open source autopilot system
Raspberrry Pi 4	is a small lightweight single-board computer
Vicon	is motion capture system, that tracks the position of objects in 3D space. By placing reflective markers on objects, multiple cameras can detect and reconstruct the position.
XBee	is a wireless connectivity module for serial communication

Acronyms

CSV	Comma Separated Values
DShot	Digital Shot
EKF	Extended Kalman Filter
ESC	Electronic Speed Controller
FC	Flight Controller
GPS	Global Positioning System
IGE	In-Ground-Effect
IMU	Inertial Measurment Unit
MAV	Micro Aerial Vehicle
MLP	Multi Layer Perceptron
ODE	Ordinary Differential Equation
OGE	Out-of-Ground-Effect
PID	Proportional-Integral-Derivative
PWM	Pulse Width Modulation
RAPTOR	Rapid Aerial Pick-and-Transfer of Objects by Robots
RPM	Rotations per Minute
SINDy	Sparse Identification of Nonlinear Dynamics

1 Introduction

1.1 Problem

Quadcopters have a wide range of applications and have proven to be precise and reliable in aerial flight maneuvers. Their simple rigid body dynamics models are used as a foundation to design most quadcopter flight controllers. Aerodynamic effects are not modeled and regarded as disturbances that the controller compensates. When the quadcopter flies maneuvers close to the ground, the magnitude of these disturbances gets bigger. This phenomenon is generally known as the ground effect and has been studied by many scientists since the beginning of flying machines. While it has been understood well for airplanes and helicopters, the emergence of quadcopters in the last decade has shed new light on the topic. Due to the more complex design of quadcopters, the disturbances are also more complicated.

1.2 Application

The RAPTOR focus project developed a quadcopter system, that can pick up objects in a rapid swooping motion(Appius et al., 2022). The current flight controller does not account for aerodynamic effects, which makes the system inaccurate near ground in the crucial phase of pickup. With the current control architecture, pick up is only possible by manually altering the position set points to compensate for ground effect disturbances. Without an improved controller, the system is unusable for real world applications. Figure 1 shows the problematic system behavior with the current controller.

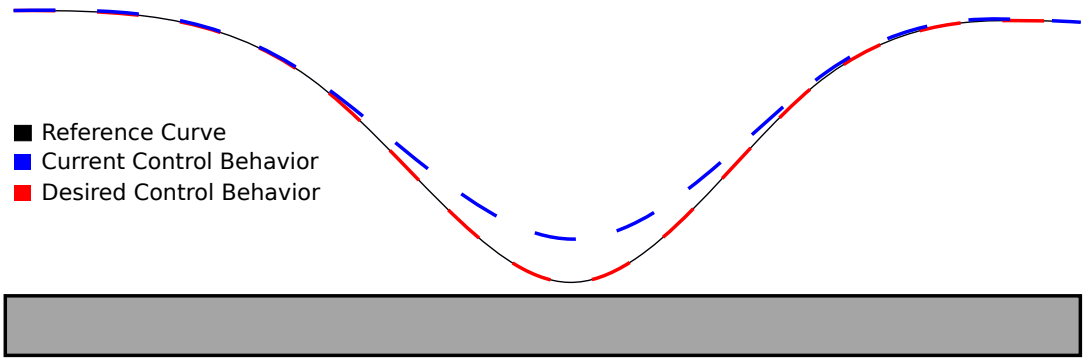

Figure 1: The problematic ground effect behavior as well as the desired behavior during a swoop.

2 Related Work

2.1 Ground Effect Studies

There is a rich literature on the influence of ground effect on various aerial vehicles. Cheeseman et al. (1955) studied the ground effect of a single rotor helicopter. He built the foundation for rotorcraft ground effect studies, by proposing equation 1. It describes the lift contribution of a helicopter with zero forward speed in the presence of ground effect. The model describes the fraction of the thrusts the quadcopter experiences between In-Ground-Effect (IGE) versus Out-of-Ground-Effect (OGE) flight.

$$\frac{T_{IGE}}{T_{OGE}} = \frac{1}{1 - \left(\frac{R}{4z}\right)^2} \tag{1}$$

Hayden (1976) proposed another experimental approximation for helicopters, which can be seen in equation 2.

$$\frac{T_{IGE}}{T_{OGE}} = \left(0.9926 + \frac{0.03794}{(z/2R)^2}\right)^{2/3} \tag{2}$$

The Cheeseman and Hayden equations describe the ground effect for single rotor aircraft, which provides a basis for the more complicated multirotor ground effect study.
Del Cont Bernard et al. (2017) showed experimentally, that the ground effect contribution is different for quadcopters and helicopters. The thrust of four isolated motors is different than the thrust produced by a quadcopter. The rotor wakes have an influence on each other and cannot be treated as independent.
Similar findings were made by Sánchez-Cuevas et al. (2017) who proposed equation 3 to describe the thrust contribution of ground effect as a function of the propeller radius R, the altitude z, the quadcopter arm length d and the experimental constant K_b that was determined to be around 2.0.

$$\frac{T_{IGE}}{T_{OGE}} = \frac{1}{1 - (\frac{R}{4z})^2 - \frac{R^2 z}{\sqrt{(d^2+4z^2)^3}} - \frac{R^2 z}{2\sqrt{(2d^2+4z^2)^3}} - \frac{2R^2 z}{\sqrt{(d^2+4z^2)^3}} K_b} \tag{3}$$

Figure 2 shows a plot of the mentioned compensators. The plot shows a clear difference between the compensators that were originally proposed for helicopter and the compensators proposed for quadcopters. It is also notable that all models approach infinity as the altitude goes towards zero.

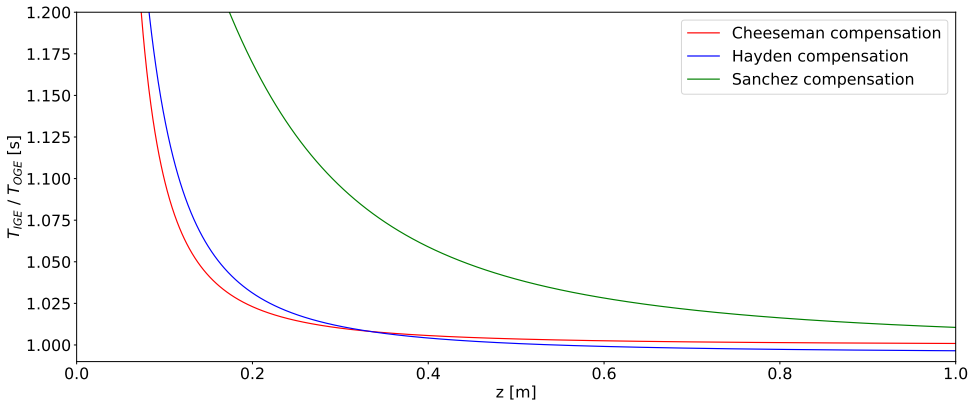

Figure 2: Thrust fraction versus altitude plot of the three static ground effect models mentioned in the related work section.

2.2 Ground effect with forward speed

Besides the model in equation 1, Cheeseman et al. (1955) also introduced a model for helicopters moving forward. This model is very similar to the static model, but includes a term that accounts for the contribution of forward speed. When a forward speed of 0 is used, the static model from equation 1 is recovered.

$$\frac{T_{IGE}}{T_{OGE}} = \frac{1}{1 - \left(\frac{R}{4z}\right)^2 \cdot \left(\frac{1}{1+\left(\frac{V}{v}\right)^2}\right)^2} \tag{4}$$

Equation 4 shows the mathematical expression of the model, where V is the forward speed of the helicopter and v is the down wash speed of the air that goes through the rotors while hovering. It can be calculated using equation 5, where T_{hover} is the hover thrust of one propeller which can also

be calculated using newtons second law with a fourth of the quadcopter mass and the gravitational acceleration.

$$v = \sqrt{\frac{T_{hover}}{2\rho_{air}\pi R^2}} = \sqrt{\frac{\frac{m_{quadcopter}}{4} \cdot g}{2\rho_{air}\pi R^2}} \tag{5}$$

Kan et al. (2019) studied the behavior of quadcopters in ground effect with forward speed. Using experimental data from two quadcopters, they came up with the model seen in equation 6, which is valid for speeds that satisfy $\frac{V}{v} < 1.2$. In the case of my quadcopter, that translates to speeds lower than 7.0 m/s.

$$\frac{T_{IGE}}{T_{OGE}} = \frac{1 - \frac{3R}{25z}}{1 + \left(\frac{3}{50}\frac{V}{v}\right)^3} \tag{6}$$

Figure 3 shows a plot of the compensators that also account for forward speed. It is notable that the difference of the model caused by a change in speed is rather small when compared to the difference between the two models. Another interesting observation is that the models change in opposite directions as the speed increases. While the Cheeseman model gets steeper with higher speeds, the Kan model flats out.

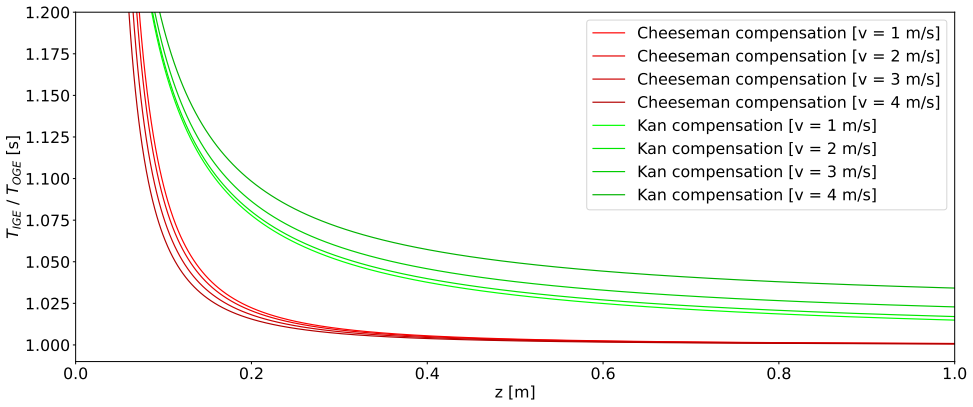

Figure 3: Thrust fraction versus altitude plot of the two dynamic ground effect models mentioned in the related work section. Due to the dependence on speed, that curves have been plotted for a selection of four forward speeds.

2.3 Control Strategies

To combat the ground effect, different approaches have been used by past research teams. Lee et al. (2021) used the sliding mode control technique to derive a stable controller that improved control performance in ground effect. Bartholomew et al. (2015) used a learning based approach to predict the ground effect by using depth images taken on the quadcopter. This resulted in a better control performance for some commonly shaped objects. He et al. (2019) used a nonlinear disturbance observer and a feed forward compensator to reduce the tracking error in ground effect by 23 %. Danjun et al. (2015) introduced a correction coefficient to equation 1 and used a ground effect compensator in their controller for a faster landing. Xuan-Mung et al. (2020) also used the equation 1 to improve landing performance.

2.4 Machine learning Approaches

Other research groups have used a machine learning approach to model ground effect. Shi et al. (2019) combined a nominal dynamic model and a deep neural network to learn the ground effect

interactions in a black-box manner. They used it to design a nonlinear feedback controller. This resulted in improved control performance during landing. Fricke et al. (2021) used a black-box Multi-Layer Perceptron (MLP) model to compensate for the additional lift near ground. They could achieve a softer landing compared to a standard PID controller.

2.5 SINDy

To efficiently counteract the ground effect, a model of the disturbance has to be identified. There are different approaches that can be used to identify unknown system dynamics. First principles modeling results in interpretable and generalizable models, but is often not applicable in complicated systems. In contrast, most machine learning modeling results in black box models that are not interpretable nor generalizable. A great contribution was made by Brunton et al. (2016) who introduced Sparse Identification of Nonlinear Dynamics (SINDy). A machine learning algorithm, that produces sparse dynamical models with minimal degree of freedom. The algorithm often finds interpretable and generalizable system dynamics. The method also allows for physical constraints to be imposed during optimization. SINDy has a proven track record and has been improved many times since its introduction. All these properties make SINDy the ideal choice to analyze the ground effect on quadcopters. The method is also easily usable thanks to the contribution by de Silva et al. (2020) and Kaptanoglu et al. (2022) who implemented the SINDy method as a python class with a profound online documentation.

3 Methodology

3.1 System Architecture

The RAPTOR quadcopter is used to investigate the ground effects on quadcopters. Additionally it is used to deploy the new controller and validate the found model. The quadcopter is built with the following components:

Pixhawk 4 The Flight Controller (FC) takes care of stabilizing and controlling the quadcopter. It runs the PX4 autopilot firmware which provides numerous features for quadcopter control.

Raspberrry Pi 4 The Raspberrry Pi 4 is used as an onboard companion computer. The primary task is forwarding the motion capture data to the Flight Controller. It can also be used to customize parts of the control loops that run on the FC.

XBee The XBee is used as a wireless connection between the Pixhawk 4 and the ground station.

Ground Computer The ground computer runs resource intensive calculations as well as code that requires user interaction. It can send high level commands directly to the flight controller over the Xbee or to the raspberry pi over wifi. Having two ways of communicating with the drone has several advantages. If one connection breaks down, the program on the ground computer can send an emergency landing command to the quadcopter over the remaining connection.

Vicon The motion capture system detects the quadcopters position which is critical for holding and tracking position commands. It is used as an alternative to the Global Positioning System (GPS) that quadcopter uses in outdoor flights.

Other The RAPTOR quadcopter has more components for tasks like grasping objects which are not of great importance in this thesis. More information about the system can be found in Appius et al. (2022).

All these components allow the system to fly complex missions, like grasping objects. Figure 4 shows a sketch of the system with all its components and their communication methods.

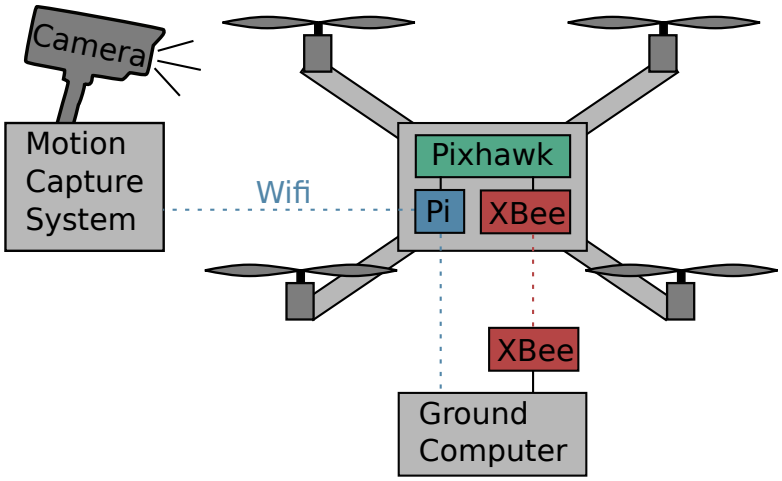

Figure 4: The RAPTOR quadcopter and its components.

3.2 Control Architecture

The PX4 autopilot system uses four cascaded PID loops to control the quadcopter. It also runs an Extended Kalman Filter (EKF) for state estimation that receives information from the motion capture system as well as the Inertial Measurment Unit (IMU). Figure 5 shows the control loops and the other key components. The following list provieds a detailed description of all the components.

Position and Velocity Controller These controllers manage the translational dynamics. A position reference is provided to the position controller which calculates a reference velocity that is fed into the velocity controller. There, a reference acceleration is determined.

Conversion to Angles Because a quadcopter can only accelerate into the direction that it is currently facing, a reference acceleration vector needs to be converted into a rotation and an acceleration in quadcopter direction. This is done by projecting the acceleration vector onto the normal vector of the quadcopter. The required orientation is found by turning the quadcopter into the direction of the reference acceleration. The reference yaw is also considered when calculating the setpoint orientation. After this conversion, the result is a reference thrust as well as a reference orientation of the quadcopter.

Angle and Angular Rate Controller The reference orientation goes into the angular controller which results in a desired angular rate. The angular rate controller calculates the angular accelerations, which are proportional to torques acting on the quadcopter. The angular accelerations and the thrust go further into the mixer.

Mixer The mixer knows about the motor configuration of the quadcopter and can convert thrust and torque commands into motor commands. The signals of the mixer go to the ESCs using the DShot protocol which causes the motors to spin with the desired speed.

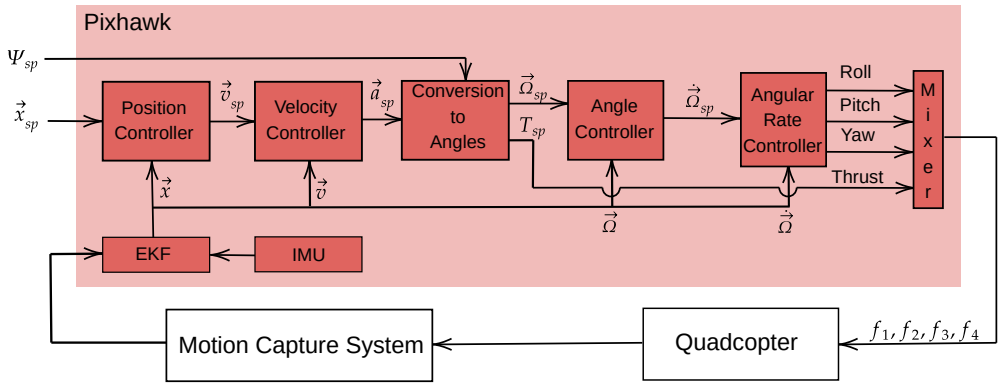

Figure 5: The PX4 control loop that runs on the Pixhawk 4 flight controller.

To include a ground effect model into the controller, a part of the loop has to be modified. It is possible to change the control loop directly on the Pixhawk 4 flight controller, but it takes a lot of experience and knowledge about the PX4 autopilot firmware. Therefore I redeployed a part of the existing control loop on the companion computer, which allowed an easy integration of a ground effect compensator. The new control architecture can be seen in figure 6. The position and velocity controller as well as the conversion to angles is a recreation of the PX4 control architecture seen in figure 5. The conversion to angles was inspired by the work of Faessler et al., 2017 and Faessler et al., 2018, which provided me with the necessary theory and mathematics to implement the controller on the onboard computer. The new controller has the possibility to modify the thrust and angle signals before they enter into the PX4 control loop for angle and angular rate control. A ground effect model runs on the raspberry and provides a thrust correction that can be added to the thrust command before sending it to the Pixhawk 4. The communication between the Pixhawk 4 and the Raspberrry Pi 4 is done over a serial connection using the MAVLink messaging protocol. A pseudocode of the quadcopter controller that runs on the Raspberrry Pi 4 can be found in appendix A.

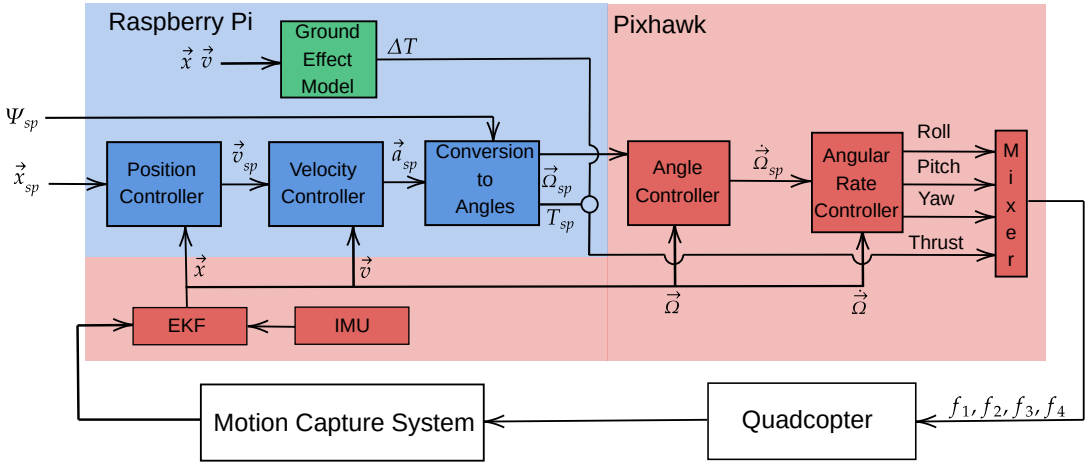

Figure 6: The PX4 control loop with a partial replacement on the Raspberrry Pi 4. The ground effect model, highlighted in green, is the new feature this controller adds in comparison to the standard approach. The other components in blue are re-implemented features of the PX4.

3.3 Thrust-Throttle Relation

The control algorithm calculates a desired thrust in Newton. The Pixhawk Flight controller expects normalized Thrust values. Due to a bad data sheet from the motor manufacturer that does not contain all the required information, the motors needed to be tested on a test stand. The stand consists of a simple setup that can be seen in figure 7. The test stand works like a lever, where the thrust force is transferred from the spinning rotor to a kitchen scale that can measure the force. This allowed me to find a relation between the motor inputs and the resulting thrust. Using a linear least squares regression on the measured data points, an expression for the quadcopter controller was found. By using the ESCs described in section 3.4, the RPM values of the motors could also be logged during the experiment. With this new information, an additional relation between the RPM and the thrust was found. Blade Element Theory states that there is a quadratic relation between the thrust and the rotational speed of a propeller. Using this information, I could easily fit a curve into the measured data points as seen in figure 8. This relation is very important to reconstruct the thrusts of each motor during a flight based on the RPM measurements.

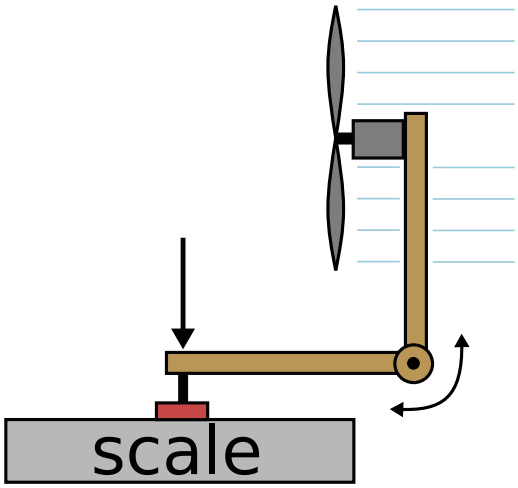

Figure 7: Test stand that was used to determine the thrust relation.

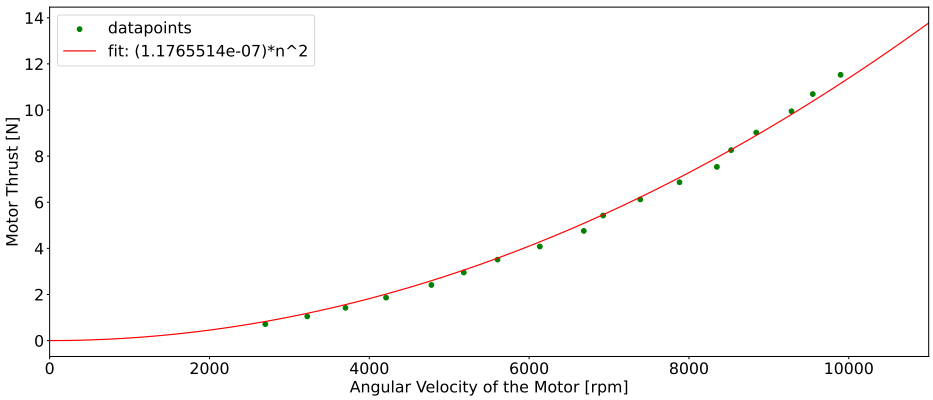

Figure 8: Datapoints measured with the setup in figure 7 and a quadratic fit.

3.4 Electronic Speed Controller (ESC)

For a successful system identification of the closed-loop quadcopter behavior, the outputs of the flight controller need to be measured, which can be achieved by measuring the motor speeds. With the information gained from the thrust-throttle-rpm test stand in section 3.3, the thrust produced by each motor can be calculated for each point in time during a mission.
To actually measure the rotational speed of each motor, the ESCs had to be replaced. There are different ways for Flight Controllers to communicate with their ESCs. The simplest way is sending a PWM signal to the ESC which contains the thrust information for the motor. A newer and faster way is the Digital Shot (DShot) protocol, which is based on a serial communication. DShot supports ESC telemetry, which is a way for the ESC to send back information about the motors to the flight controller. This is achieved by an additional serial connection back to the flight controller. The DShot message to the ESC contains a request for telemetry data which is then sent back. A conceptual sketch of the setup can be seen in figure 9.

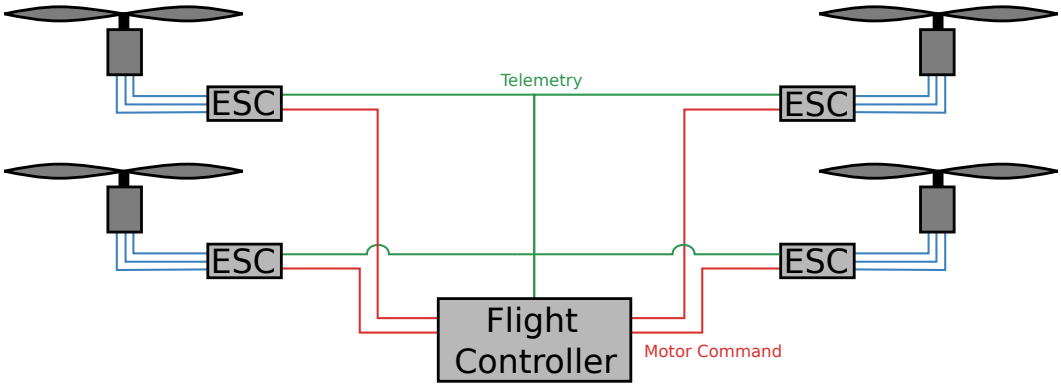

Figure 9: DShot based quadcopter setup for motor speed measurment

3.5 Data collection

A quadcopter flight can only be analyzed on ground effect influence if enough data was collected during the flight. A lot of data can be accessed through the MAVSDK *Telemetry* class, which provides direct access to the state estimation of the flight controller. An Extended Kalman Filter (EKF) runs on the flight controller that is fed sensor information from the IMU as well as motion capture data. Among many other states, the following are of great interest for later flight analysis: Position, Velocity, Acceleration, Orientation, Angular Velocities. While this already great, information about the system outputs are crucial to perform useful closed loop system analysis. By using the ESC's described in section 3.4, the angular velocities of each motor, hence the flight controllers outputs are measured. Accessing this information from the C++program is not trivial, because there is no MAVSDK class for this. Luckily, it was possible to log motor speeds, by directly accessing the underlying MAVLink messages using the MAVSDK *MavLinkPassthrough* class. More information and access to the captured flight data can be found in appendix B.

3.6 Metrics

Metrics are defined to quantify the Flight performance, which is needed for a methodological comparison between different control strategies. By only looking at the plots, it is not possible to unambiguously determine a change in flight behavior. A mathematical definition that describes the "goodness" of the flight performance yields a clear unbiased metric. I ended up choosing the following two metrics M_A and M_B to rate the flight behavior.

M_A Comparison with the reference curve: This general approach can be used to determine various control performances. It is not specifically made to rate ground-effect behavior. The

metric is defined as the absolute integral error during the measured time. The mathematical definition can be seen in equation 7.

$$M_A = \int_{T_0}^{T_1} x(t) - x_{ref}(t)\, dt \tag{7}$$

M_B Comparison with high altitude behavior: While M_A is useful to quantify general control behavior, it is not formulated specifically to rate the influence of ground effect disturbances. Therefore, the better suited M_B metric is defined as the absolute integral error between the trajectory and the same trajectory in a higher altitude. To apply this metric, the mission has to be flown at both the normal altitude where ground effect is present as well as a higher altitude. The high-altitude trajectory is lowered to the same height as the regular trajectory so that they are comparable. With this metric, the control errors that are present regardless of ground effect or not considered. If M_B becomes zero, the disturbances due to ground effect have been compensated completely. The mathematical definition can be seen in equation 8.

$$M_B = \int_{T_0}^{T_1} x_{IGE}(t) - (x_{OGE}(t) - \Delta z)\, dt \tag{8}$$

These Metrics can be applied to any desired coordinate that needs to be investigated. To account for random disturbances and noise, the metrics are usually applied to the average value over many missions, which also allows the calculation of a standard deviation. A graphical interpretation of M_A and M_B on an example step response can be seen in figure 10. The Values for this example are $M_A = 1.972\,\mathrm{m\,s}$ and $M_B = 1.043\,\mathrm{m\,s}$ where as expected M_B is the smaller value. To have a meaningful comparison between two flights, it is important to calculate the metric over the same time interval $\Delta T = T_1 - T_0$. Otherwise the error of the trajectory that was measured over a longer time interval will unjustly be larger.

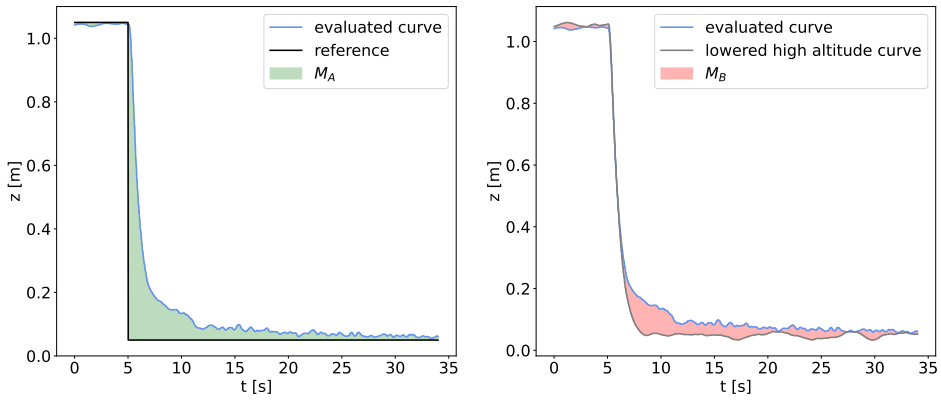

Figure 10: Graphical interpretation of M_A and M_B on a step response in the z-coordinate

3.7 Ground effect influence from quadcopter dynamics

The initial idea of the thesis was to discover the ground effect influence in the quadcopters state space representation. By applying the SINDy algorithm on flight data, a set of ODEs describing the underlying dynamics of the data can be discovered. I anticipated that these ODEs would consist of the original quadcopter rigid body dynamics as well as residual terms that I could then classify as ground effect influence. After conducting some initial experiments and a few tries with the SINDy algorithm, I recognized that the method was bound to fail. The data was too noisy and the aerodynamical disturbance were too big in compairison to the measured variables. The SINDy did not converge and could not find a satisfying sparse representation of the data. This made me rethink my thesis' goals and I realized that the inital approach was not ideal for the problem. By

trying to identify ground effect influence over the state space equations of the dynamical system, I was taking a big detour to my goal. Therefore I changed my plan and went with an alternative approach described in section 3.8.

3.8 Ground effect influence from direct flight data regression

Instead of finding the dynamics that govern the captured flight data, it is also possible to directly investigate the captured data. The following subsections explain the methods used to expose the ground effect influence in the flight data as well as the later analysis of the data.

3.8.1 Data processing

The discovered ground effect model should be comparable to the models found in the related work, which can be achieved by finding equations with a similar structure. Equation 9 shows the typical structure of the already existing ground effect models. The goal of the data analysis is finding a relation between the fraction of the thrusts that the quadcopter produces outside and inside the ground effect region.

$$\frac{T_{IGE}}{T_{OGE}} = f(z,\ v_x,\ v_y,\ v_z, \dots) \tag{9}$$

A simple way of collecting this kind of data is by flying the same trajectories in the ground effect zone as well as outside of it. For a better understanding, this core idea is illustrated in figure 11. The collected data is split into two parts, an OGE part and an IGE part. A division of these two gives us the left hand side of equation 9. The right hand side can then be found by using the machine learning framework on the other logged variables.

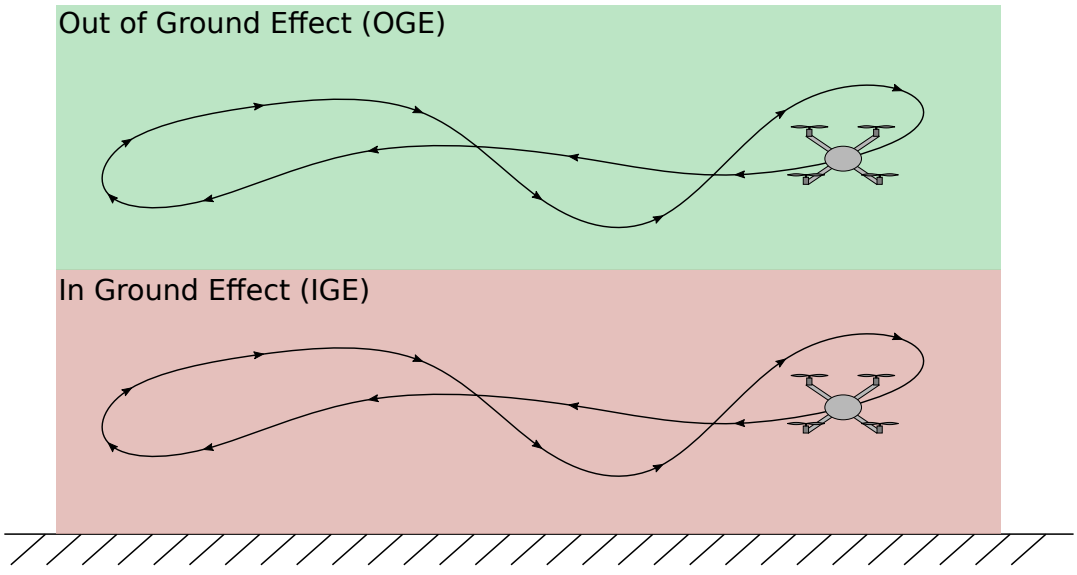

Figure 11: Flying the same trajectory on different heights to collect data that can be processed into a useful form for later machine learning analysis.

3.8.2 SINDy Modification

The SINDy framework was originally designed to discover a sparse nonlinear system of Ordinary Differential Equation (ODE)s from empirical data. The original idea of this thesis was to discover

the ground effect contribution in the state space equations of the quadcopter dynamics. As described in section 3.7, this approach turned out to be impractical and was therefore discarded. The SINDy framework still seemed useful because I still wanted to find a sparse nonlinear representation of my data. Instead of finding an ODE, I wanted to find a sparse analytical expression describing my flight data. Luckily, the pySINDy library allows the user to change the left hand side of the equation with custom data. As seen in equation 10 I replaced the derivative of the ODE with the ground effect thrust fraction.

$$\begin{aligned} \dot{z} &= f(z, v_x, v_y, v_z, \dots) \\ &\Downarrow \\ \frac{T_{IGE}}{T_{OGE}} &= f(z, v_x, v_y, v_z, \dots) \end{aligned} \tag{10}$$

3.8.3 Machine learning pipeline

Because the non-linearity is not known, the SINDy framework is used to find the form of the nonlinear function. To perform the fit, a library of nonlinear functions as well as data has to be provided. The nonlinear functions are applied to the data which results in the matrix $\Theta(x)$. In a next step, w is optimized to find a sparse solution to equation 11.

$$y = \Theta(x)w \tag{11}$$

The optimization searches for an optimum between sparsity and error, minimizing the cost function seen in equation 12 where λ is the sparsity parameter. The optimizer works by computing a series of linear least square fits while masking out values below a certain threshold in the w matrix. This results in a sparse nonlinear fit of the data.

$$L(w) = \underbrace{||y - \Theta(x)w||}_{Error} + \underbrace{\lambda||w||_0}_{Sparsity} \tag{12}$$

The big limitation with this method is that the fitting parameters can only enter linearly into the optimization. Therefore a second nonlinear curve fit is performed on the data, where the ansatz function is inspired by the result of the SINDy fit. This two-step regression allows me to find a compact equation representing the data, without having to know about the possible non-linearities beforehand. The whole process is illustrated in figure 12.

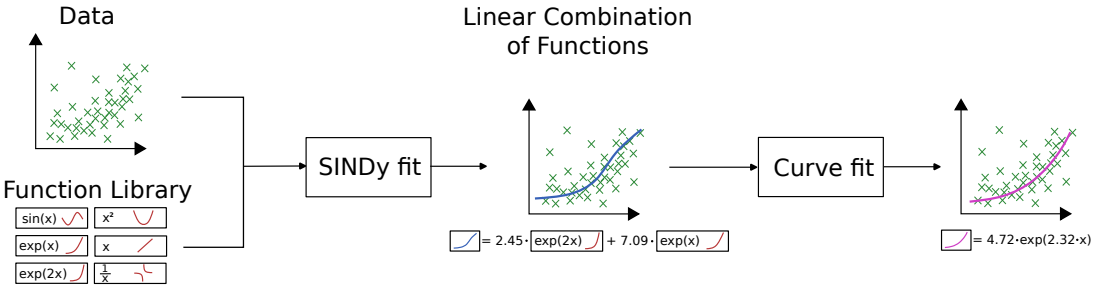

Figure 12: The machine learning pipeline used in this thesis. The information gained from the SINDy fit is used to fit a curve into the data.

4 Experiments and Results

All experiments in this thesis can be grouped in two main categories. The first category is described in section 4.1 and contains all experiments that were used to generate data for later ground effect influence analysis. The second category is found in section 4.3 and contains all experiments that

were used to validate the performance of the existing and newly found ground effect models. All data was collected using a Vicon motion capture system. According to Vicon Motion Systems Ltd. UK (2020), the standard deviation of the measurement is 0.198 mm, which is significantly smaller than all measurements conducted in this thesis. Therefore, the error caused by the inaccuracy of the motion capture system is insignificant and not treated any further.

4.1 Data collection trajectories

These trajectories were used to collect data that would then be analyzed on ground effect influence. The following three subsections describe those different trajectories in detail.

4.1.1 Hovering on different heights

The most simple way to find a ground effect influence on a quadcopter is by measuring the hovering thrust on different heights. A plot of the altitude and the measured total motor thrust during this experiment can be seen in figure 13. By just looking at the plots from this experiment, there is a strong indication visible that the height and the thrust are somehow related. A later machine learning analysis will prove this assumption to be true.

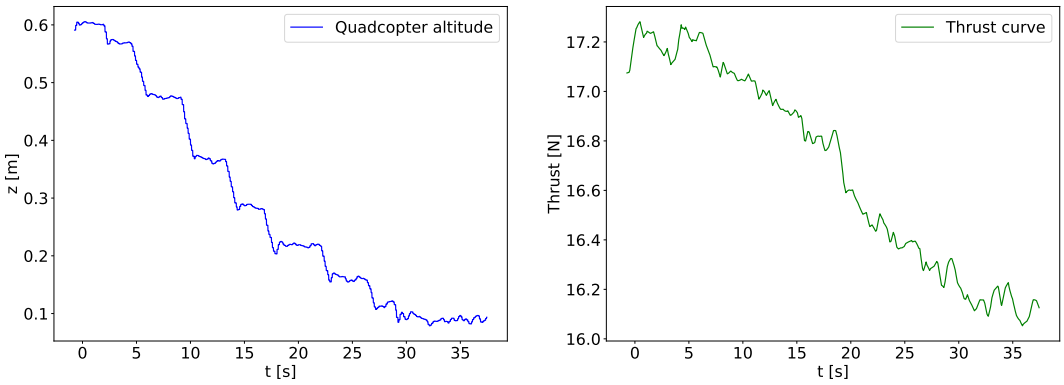

Figure 13: The altitude plot as well as the thrust provided by the quadcopter during the hovering maneuver.

4.1.2 Hovering on different heights while moving forward

While the trajectory described in section 4.1.1 does a good job at exposing some ground effect disturbances it possibly only shows an excerpt of the whole ground effect disturbance. That is why a second trajectory is proposed where the quadcopter does essentially the same thing but while moving forward at various speeds. A plot of the altitude and horizontal speeds during this maneuver can be seen in figure 14. The goal was to achieve the horizontal speeds of 0.5, 1.0, 1.5 and 2.0 $\frac{m}{s}$ on different heights. The size of the motion capture room in which the experiments were conducted posed a serious limitation on the forward speeds. Once the quadcopter had accelerated to a higher speed, he was already approaching the wall on the other side of the room and had to decelerate. Due to this restriction, it was not possible to test higher speeds and it was also difficult to capture high quality data with the rather lower speeds. The targeted velocities are visible as the peaks of the velocity plot in figure 14. There are always two peaks visible because the quadcopter flew across the room in one direction and the back.

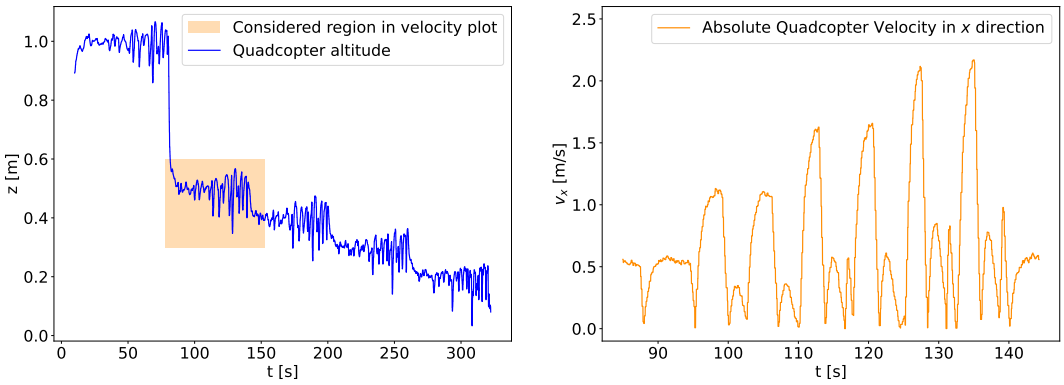

Figure 14: The altitude plot as well as the velocity plot of the quadcopter during the flight.

4.1.3 Moving vertically on different heights

An influence of the vertical speed on ground effect disturbances seamed possible and was investigated. Collecting data with different vertical speeds while being in the ground effect region was very difficult. If the vertical velocities were too high, the quadcopter would either crash into the ground or leave the thin zone where ground effect is present. The only way of collecting data was by either doing low altitude vertical oscillations, or by flying into the ground effect zone with different vertical speeds and braking just before hitting the ground. The oscillation trajectory can be seen in figure 16 and the flying into ground effect zone with different speeds can be seen in figure 15. The red and green box emphasize that the same trajectory was flown both Out-of-Ground-Effect (OGE) and In-Ground-Effect (IGE).

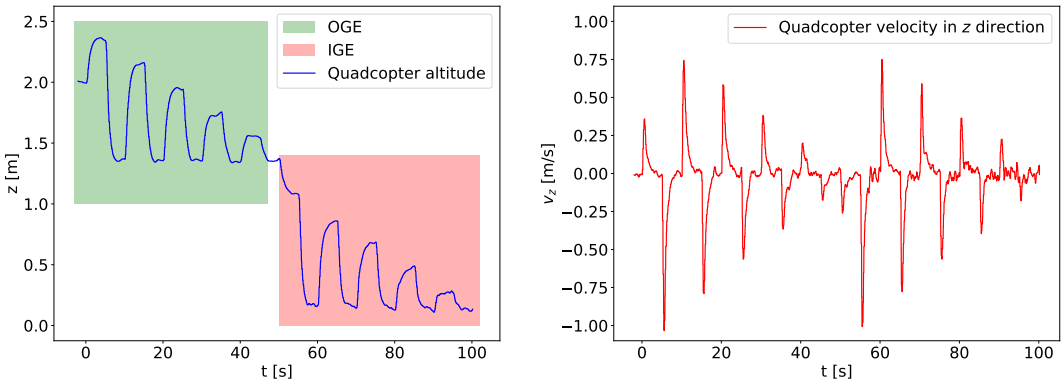

Figure 15: The altitude plot as well as the vertical velocity plot of the quadcopter during the flight.

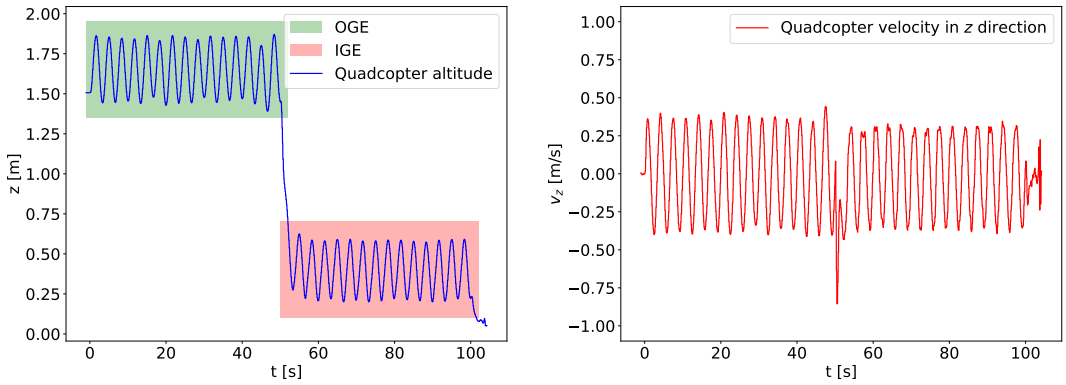

Figure 16: The altitude plot as well as the velocity plot of the quadcopter during the oscillation flight.

4.2 Discovered Ground Effect Models

The trajectories described in section 4.1 were used to find different ground effect models. The data was analyzed using the approach described in section 3.8.3.

4.2.1 Static model

Using only hovering data, a relation between the quadcopter altitude and the thrust had to be identified. In the case where there are only two variables, a 2D-plot which can be seen in figure 17 can be used to visually find a correlation between the two variables. To find the nonlinear correlation analytically with SINDy, the function library in table 1 was used. The SINDy algorithm converged with the model seen in equation 13. After an initial look at the equation, it could be observed that the algorithm converged to solutions where exponential terms were dominating. Due to the limitation of SINDy to only fit linear combinations of nonlinear terms as described in section 3.8.3, a second fit was made. A SINDy-fit inspired ansatz of the form $a \cdot e^{bz} + c$ was used to fit the data. This resulted in equation 14, which is also indicated in the plot in figure 17.

Function class	$f(x)$
Exponential	$exp(-x),\ exp(-2x),\ exp(-3x),\ \ldots$
Identity	1
Polynomial	$x,\ x^2,\ x^3,\ \ldots$
Inverse Polynomial	$\frac{1}{x},\ \frac{1}{x^2},\ \frac{1}{x^3},\ \ldots$

Table 1: Library of non-linearities that were used in the SINDy algorithm.

$$\frac{T_{IGE}}{T_{OGE}} = -1.404 \cdot e^{-3z} + 4.750 \cdot e^{-2z} - 6.942 \cdot e^{-z} + 4.719 - 1.948 \cdot z + 0.250 \cdot z^2 \tag{13}$$

$$\frac{T_{IGE}}{T_{OGE}} = 0.114 \cdot e^{-5.388z} + 1.028 \tag{14}$$

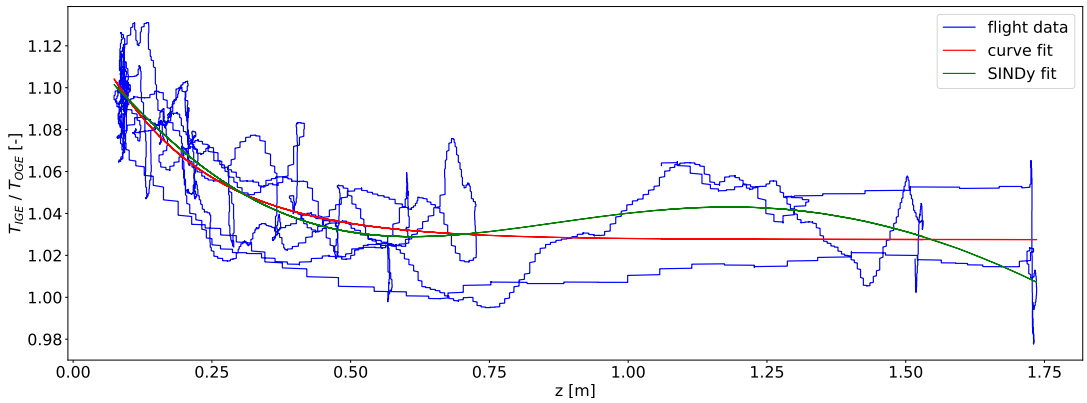

Figure 17: Hovering flight data with height plotted against the quotient of T_{IGE} and T_{OGE}. Furthermore, the SINDy and the normal curve fit are shown.

4.2.2 Forward velocity model

Besides looking at height, the forward speed of the quadcopter could also play an important role on the ground effect influence. Therefore, that data described in section 4.1.2 was used to find another model. The presence of different forward speeds in the data should yield a model of the from $\frac{T_{IGE}}{T_{OGE}} = f(z, v_x)$. To find the model, the data was given to the SINDy algorithm for a sparse regression. The SINDy algorithm discovered the expression seen in equation 15. As seen in the mathematical expression, there is no contribution of the forward speed to the ground effect. Also by looking at the plot on the right side of figure 18, there does not seem to be any relation present in the data. This does not mean that forward speed does not affect quadcopter ground effect at all. It only means that there is no correlation in the measured data, which is possibly due to the small speeds of only 2 m/s.

$$\frac{T_{IGE}}{T_{OGE}} = 0.253 \cdot e^{-3z} - 0.581 \cdot e^{-2z} + 1.450 \cdot e^{-z} + 0.563 \cdot z \tag{15}$$

By taking the information from the SINDy-fit, that the curve is described by exponentials, a curve fit can be performed which brings us to the much simpler term seen in equation 16. We observe that the discovered model is very similar to the one found in subsection 4.2.1.

$$\frac{T_{IGE}}{T_{OGE}} = 0.147 \cdot e^{-4.368z} + 0.988 \tag{16}$$

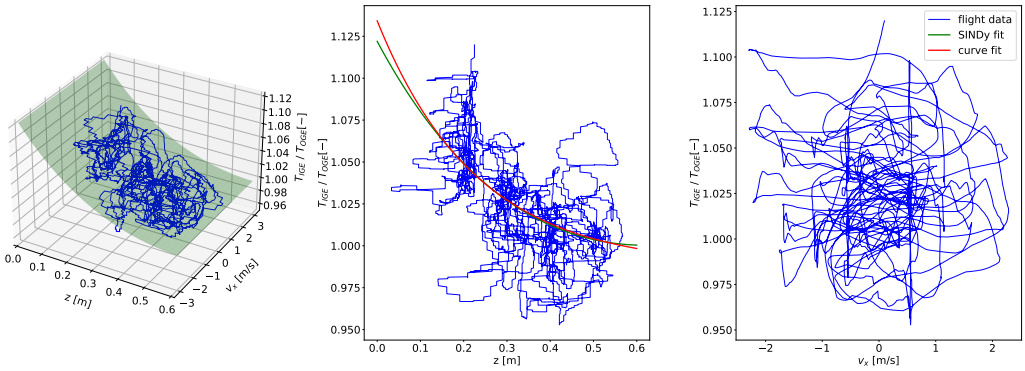

Figure 18: Flight data where the quadcopter flies at several forward velocities. The 3D-Plot shows the flight path and the SINDy curve fit. The other two plots show the z-dependence and the v_x-dependence isolated.

4.2.3 Vertical velocity model

The last investigated variable is the vertical speed. The same machine learning procedure was applied to the data collected with the trajectories described in subsection 4.1.3. Both oscillation and downward speed trajectories were analyzed. As seen in figure 18 there is no correlation visible between the vertical speed v_z and the ground effect contribution. Again, this does not mean that there is no influence of downward speed, but only that it is possibly not present in the data set. The model found by the SINDy algorithm is also only depended on the z variable, but inferior to the model found in section 4.2.1 and 4.2.2 because the dataset is a lot smaller.

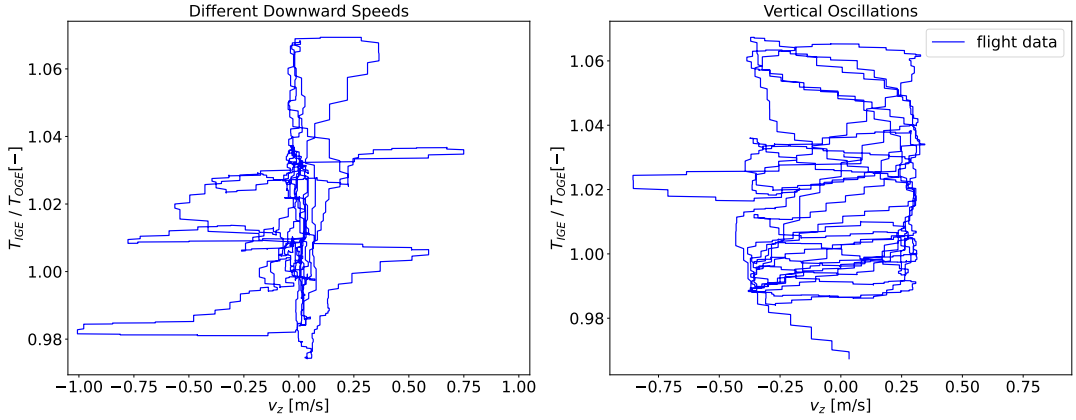

Figure 19: Influence of v_z on the ground effect of both trajectories described in section 4.1.3.

4.3 Model validation trajectories

These trajectories were used to validate the found models as well as the related work models. In the following section, the model that is referred to as "Appius Compensator" is the one that can be seen in equation 14. It is very similar to the model in equation 16 but more accurate due to the simpler flight path that was flown to generate the data.

4.3.1 Static Step response

The influence of the ground effect can easily be seen when approaching the ground with the quadcopter. The simplest way to show the behavior is by performing a step response along the z-axis. To benchmark the existing ground effect compensators with each other and my model, they were deployed using the control architecture described in section 3.2. To get meaningful results, each compensator was tested five times. The experimental data was used to calculate average trajectories as well as standard deviations, which can be seen in figure 20.

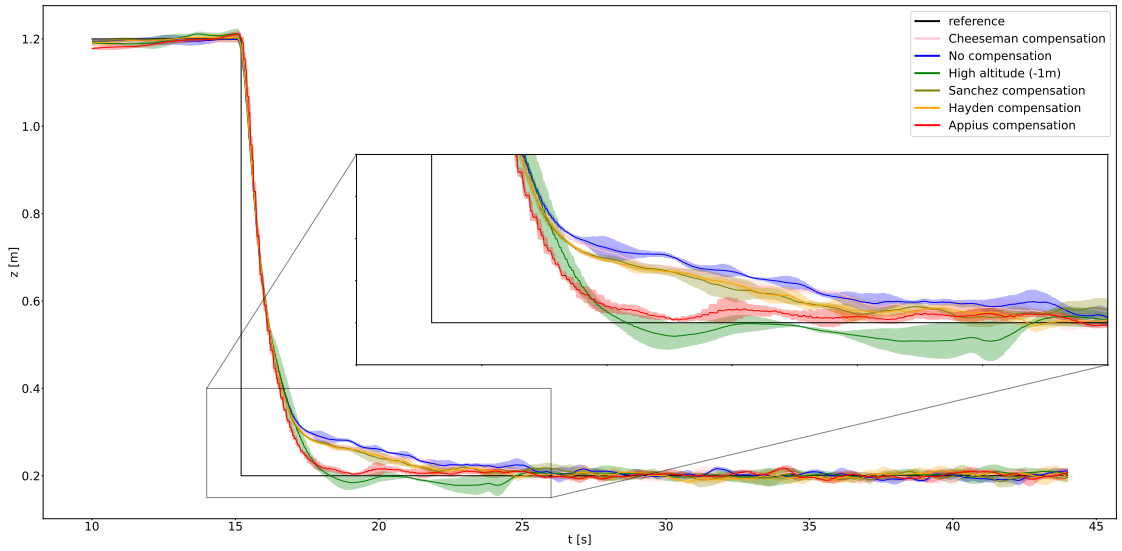

Figure 20: Control performance of different ground effect compensators during a z-axis step response

To quantify the performance even better, the metrics defined in section 3.6 were applied to the data. The results can be seen in table 2. By looking at the data, an improvement from no-compensation to the related work compensators is clearly visible. Although there is an improvement in the performance, a comparison with the behavior in high altitudes shows, that the overall performance is still very weak. On the other hand, we can observe that the compensator that was found by analyzing the flight data of the quadcopter, performs very good. When looking at metric M_B, my model can reduce the error by 78.5 % where as other compensators only reduce the error by 36.9 %. This marks an improvement of 212 % over the compensators found in literature.

Compensator	M_A	%-error (M_A)	M_B	%-error (M_B)
No compensation	1.441 ± 0.007	100 %	0.409 ± 0.007	100 %
Cheeseman compensation	1.36 ± 0.013	94.4 %	0.328 ± 0.013	80.2 %
Hayden compensation	1.301 ± 0.006	90.3 %	0.269 ± 0.006	65.8 %
Sanchez compensation	1.29 ± 0.017	89.5 %	0.258 ± 0.017	63.1 %
Appius compensation	1.12 ± 0.033	77.7 %	0.088 ± 0.033	21.5 %
High altitude (-1m)	1.032 ± 0.029	71.6 %	0.0 ± 0.029	0 %

Table 2: Metrics of the compensators on a step response with standard deviation

4.3.2 Step response with forward speed

As seen in subsection 4.3.1, the step response is a good trajectory to visualize and quantify ground effect influence. A more thorough look at the performance of the compensator and controller can be seen by conducting a step response wile flying forward. In this experiment, I conducted vertical steps with 0.5, 1.25, 2.0 m/s forward speed. The resulting step responses can be seen in figure 21. The calculated metrics, that are visible in table 3 show a clear improvement in the performance.

Surprisingly, the compensators with forward speed from the related work described in section 2.2 made the quadcopter unstable which resulted in a crash. Therefore these compensators could not be tested on the drone and the metrics could not be calculated. The compensator can reduce the error by 59.2 % for 0.5 m/s, 39.7 % for 1.25 m/s and 38.0 % for 2.0 m/s.

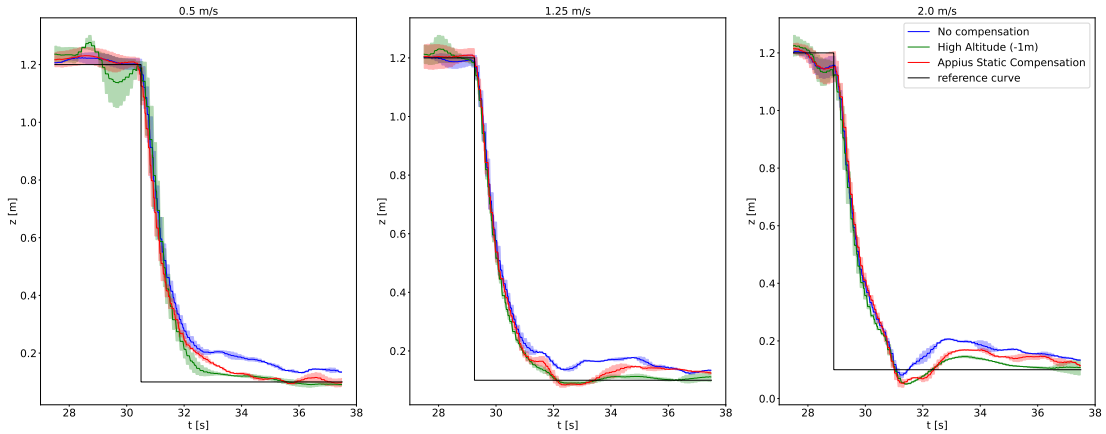

Figure 21: Vertical step responses with different forward speeds. The curves show the different controllers that were used while performing the maneuver.

M_A	0.5 m/s	1.25 m/s	2.0 m/s
No Compensator	1.219 ± 0.122	1.242 ± 0.079	1.227 ± 0.063
High Altitude	0.936 ± 0.072	0.869 ± 0.064	0.897 ± 0.06
Appius Compensator	1.023 ± 0.181	1.099 ± 0.142	1.085 ± 0.117

M_B	0.5 m/s	1.25 m/s	2.0 m/s
No Compensator	0.333 ± 0.123	0.36 ± 0.079	0.371 ± 0.063
High Altitude	0.0 ± 0.072	0.0 ± 0.064	0.0 ± 0.061
Appius Compensator	0.136 ± 0.182	0.217 ± 0.142	0.23 ± 0.118

Table 3: Metrics M_A and M_B of the compensator on a step response with different forward speeds

4.3.3 Swoop

This trajectory is used to show the generalizability of the found ground effect model and the resulting controller. This flight path is not present in the data collection trajectories described in section 4.1. As seen in figure 22, the found compensator can reach the target height with, whereas the controller with no compensation misses the target height by 3 cm. By showing this behavior, this thesis solves its initial problem that was described in section 1. The controller and the found model could now be incorporated into the RAPTOR system to reduce tracking error and therefore improve grasping success during the swooping maneuver.

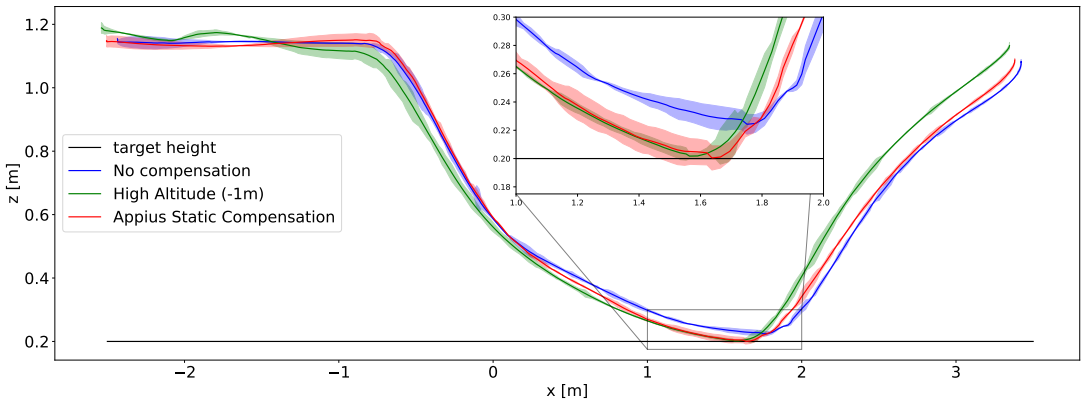

Figure 22: Swooping maneuver with different controllers.

5 Discussion

5.1 Key Results

Using a custom control architecture, I could show that older and current ground effect compensators perform very poor and do not correctly describe the aerodynamics experienced in the ground effect region. By flying various trajectories and analyzing the flight data, a better model was found, that outperformed all other compensators and significantly increased control performance near ground. By analyzing other trajectories, the influence of vertical speed as well as horizontal speed turned out to be negligible in the range of velocities that our specific quadcopter could fly. The found ground effect model was also deployed on various scenarios that were not present in the training data. Also there, the model and controller could almost completely counteract the ground effect disturbance. With that in mind, it can be said that the discovered model generalizes well and is not over fitted.

5.2 Ground Effect Model

The SINDy algorithm converged on an exponential function and not on a polynomial. The ground effect models found in literature are all rational functions. They all have a pole where the ground effect influence goes to infinity, which does not really make sense. While they describe the ground effect contributions approximately right for larger altitudes, they explode to infinity at a certain height which completely messes up the control performance. In that scenario the controller has to adjust to the wrong compensation making him essentially worse than before.

5.3 Controller Tuning

The PID-gains for the quadcopter controller were tuned in high altitude flight with no ground effect influence. This explains the nice control behavior with 0 % overshoot and an average rise time t_{90} of approximately 2 seconds for the curves flown out of ground effect. By adjusting the controller gains, the behavior in the ground effect region could be improved. However, this would then probably result in a worsened control behavior in higher altitudes which is not desired. Therefore the control gains were not changed in this thesis to prevent a worsening in the high altitude control performance.

5.4 Machine Learning Alternatives

The machine learning approach used in this thesis was designed continuously while conducting the experiments. The reader might be confused about the two succeeding curve fits, that might also be done in one step. Therefore, it has to be mentioned that there are alternatives to the approach

presented here. One might also use the system identification toolbox from MATLAB which also provides features for sparse modeling of nonlinear data.

5.5 Limitations

The found ground effect model describes the disturbance very well and also performs well in the used controller. It also generalizes beyond the situations present in the training data set. While the model works well for the quadcopter used in this thesis, it would probably fail reproduce the results on an other quadcopter architecture. Changing the weight, the motors, the propellers, the geometry or other parameters would result in different aerodynamics and therefore a different ground effect influence. While other models in literature propose general models for all multi copter architectures, my model works only on my specific quadcopter architecture. Instead of proposing a general analytical ground effect model for all quadcopters, I provide a set of flight analysis tools that allow other scientists and engineers to easily find a model using their flight data for their specific quadcopter. This approach yields better models that are custom-made for each quadcopter.
The ground effect model is still very simple and is only dependent on the altitude. In this thesis no influence of vertical and horizontal speed could be found. Given the models found in literature that found a relation between ground effect and horizontal speed, there definitely is some influence. Due to the small room size of the flight arena it was only possible to fly at fairly low speeds which hindered the collection of better and more meaningful data.
The found model only describes a thrust difference invoked by the ground effect, which might not be the full picture. While thrust is definitely the most dominant ground effect contributor other disturbances like torques could also be analyzed and counteracted.

5.6 Further Research

I proposed a novel easy-to-implement approach to tackle quadcopter ground effect disturbance. While the proposed solution works, there are still a lot of open problems. The machine learning pipeline would ideally be fused into one big SINDy-based regression instead of performing two successing regressions.
My approach could also be expanded to other aerodynamical disturbances that quadcopters experience. The partial ground effect, where the quadcopters propellers are asymmetrically affected by ground effect, could be analyzed and a compensating controller could be deployed.
The framework could potentially also be extended to cover wall-effects when the drone flies close to a wall or even to an object with an arbitrary geometry.

6 Conclusion

This thesis put other researchers and engineers who deal with ground effect disturbances in their system one step closer to an easy practical solution. I showed that existing analytical ground effect models do not perform well for all quadcopter architectures. Instead of proposing another analytical model, a machine learning approach is presented that finds high-accuracy and low-effort models for quadcopters by analyzing flight data.
Furthermore a custom controller was designed and implemented on an onboard computer. The new controller could improve performance in low altitude flight due to its knowledge about the aerodynamical disturbances.

References

Appius, A., Bauer, E., Blochlinger, M., Kalra, A., Oberson, R., Raayatsanati, A., Strauch, P., Suresh, S., von Salis, M., & Katzschmann, R. K. (2022). Raptor: Rapid aerial pickup and transport of objects by robots. *ArXiv*, *abs/2203.03018* (cit. on pp. 1, 4).

Bartholomew, J., Calway, A., & Mayol-Cuevas, W. (2015). Improving mav control by predicting aerodynamic effects of obstacles. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4826–4833 (cit. on p. 3).

Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, *113*, 3932–3937 (cit. on p. 4).

Cheeseman, I. C., D, P., Bennett, W. E., D, P., & Bennett, W. E. (1955). *The effect of ground on a helicopter rotor in forward flight* (tech. rep.). (Cit. on pp. 1, 2).

Danjun, L., Yan, Z., Zongying, S., & Geng, L. (2015). Autonomous landing of quadrotor based on ground effect modelling. *2015 34th Chinese Control Conference (CCC)*, 5647–5652. https://doi.org/10.1109/ChiCC.2015.7260521 (cit. on p. 3)

Del Cont Bernard, D., Giurato, M., Riccardi, F., & Lovera, M. (2017). Ground effect analysis for a quadrotor platform (cit. on p. 2).

de Silva, B., Champion, K., Quade, M., Loiseau, J.-C., Kutz, J., & Brunton, S. (2020). Pysindy: A python package for the sparse identification of nonlinear dynamical systems from data. *Journal of Open Source Software*, *5*(49), 2104. https://doi.org/10.21105/joss.02104 (cit. on p. 4)

Faessler, M., Falanga, D., & Scaramuzza, D. (2017). Thrust mixing, saturation, and body-rate control for accurate aggressive quadrotor flight. *IEEE Robot. Autom. Lett.*, *2*(2), 476–482. https://doi.org/10.1109/LRA.2016.2640362 (cit. on p. 6)

Faessler, M., Franchi, A., & Scaramuzza, D. (2018). Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories. *IEEE Robot. Autom. Lett.*, *3*(2), 620–626. https://doi.org/10.1109/LRA.2017.2776353 (cit. on p. 6)

Fricke, K., Nascimento, R. G., & Viana, F. A. C. (2021). Quadcopter soft vertical landing control with hybrid physics-informed machine learning. *AIAA SciTech Forum*. https://doi.org/10.2514/6.2021-1018 (cit. on p. 4)

Hayden, J. S. (1976). The effect of the ground on helicopter hovering power required (cit. on p. 1).

He, X., Kou, G., Calaf, M., & Leang, K. K. (2019). In-Ground-Effect Modeling and Nonlinear-Disturbance Observer for Multirotor Unmanned Aerial Vehicle Control [071013]. *Journal of Dynamic Systems, Measurement, and Control*, *141*(7). https://doi.org/10.1115/1.4043221 (cit. on p. 3)

Kan, X., Thomas, J. R., Teng, H., Tanner, H. G., Kumar, V. R., & Karydis, K. (2019). Analysis of ground effect for small-scale uavs in forward flight. *IEEE Robotics and Automation Letters*, *4*, 3860–3867 (cit. on p. 3).

Kaptanoglu, A. A., de Silva, B. M., Fasel, U., Kaheman, K., Goldschmidt, A. J., Callaham, J., Delahunt, C. B., Nicolaou, Z. G., Champion, K., Loiseau, J.-C., Kutz, J. N., & Brunton, S. L. (2022). Pysindy: A comprehensive python package for robust sparse system identification. *Journal of Open Source Software*, *7*(69), 3994. https://doi.org/10.21105/joss.03994 (cit. on p. 4)

Lee, J.-W., Xuan-Mung, N., Nguyen, N. P., & Hong, S. K. (2021). Adaptive altitude flight control of quadcopter under ground effect and time-varying load: Theory and experiments. *Journal of Vibration and Control*. https://doi.org/10.1177/10775463211050169 (cit. on p. 3)

Sánchez-Cuevas, P. J., Heredia, G., & Ollero, A. (2017). Characterization of the aerodynamic ground effect and its influence in multirotor control. *International Journal of Aerospace Engineering*, *2017*, 1–17 (cit. on p. 2).

Shi, G., Shi, X., O'Connell, M., Yu, R., Azizzadenesheli, K., Anandkumar, A., Yue, Y., & Chung, S.-J. (2019). Neural lander: Stable drone landing control using learned dynamics. *2019 International Conference on Robotics and Automation (ICRA)*. https://doi.org/10.1109/icra.2019.8794351 (cit. on p. 3)

Vicon Motion Systems Ltd. UK. (2020). FAQs For Accuracy. https://www.vicon.com/support/faqs/?q=how-accurate-precise-are-your-systems. (Cit. on p. 12)

Xuan-Mung, N., Hong, S. K., Nguyen, N. P., Ha, L. N. N. T., & Le, T.-L. (2020). Autonomous quadcopter precision landing onto a heaving platform: New method and experiment. *IEEE Access*, *8*, 167192–167202. https://doi.org/10.1109/ACCESS.2020.3022881 (cit. on p. 3)

Appendix A Control Loop Pseudocode

The following pseudocode shows the basic structure of the quadcopter control loop used to compensate ground effect disturbances. A full functioning code written in C++ can be found on GitHub in the aurelappius/quad_control repository.

Pseudocode 1 Quadcopter control with Ground Effect Compensation

Initialize quadcopter communication with MAVSDK
Initialize Logging
Arm quadcopter
while quadcopter != landed **do**
    $Position$, $Velocity$, $Acceleration$, $Orientation \leftarrow Telemetry$
    $Position_{Ref}$, $Yaw_{Ref} \leftarrow$ TrajectoryGenerator($Time$, $Position, \dots$)

    $Error_{P,Pos} \leftarrow Position_{Ref} - Position$ Position P controller
    $Velocity_{Ref} \leftarrow P_{Pos} \cdot Error_{P,Pos}$

    $Error_{P,Vel} \leftarrow Velocity_{Ref} - Velocity$ Velocity PID controller
    $Error_{I,Vel} \leftarrow Error_{I,Vel} + Error_{P,Vel} \cdot \Delta T$
    $Error_{D,Vel} \leftarrow (Error_{P,Vel} - LastError_{P,Vel})/\Delta T$
    $Acceleration_{Ref} \leftarrow P_{Vel} \cdot Error_{P,Vel} + I_{Vel} \cdot Error_{I,Vel} + D_{Vel} \cdot Error_{D,Vel}$

    Calculate $Orientation_{Ref}$ based on $Acceleration_{Ref}$ and Yaw_{Ref}
    Project $Acceleration_{Ref}$ on quadcopter frame z-axis from $Orientation$
    $Thrust \leftarrow$ProjectedAcceleration$_{Ref} \cdot QuadcopterMass$

    $Thrust_{Corrected} \leftarrow$ GroundEffectCompensator($Thrust$, $Position, \dots$)

    SendToQuadcopter($Thrust_{Corrected}$, $Orientation_{Ref}$)

    LogData($Position$, $Velocity$, $Acceleration$, $Orientation$)

    Wait for($20ms$) 50Hz controller frequency
end
Disarm quadcopter

Appendix B Flight Logs

All the data collected during this thesis is freely accessible for further investigation. The data is provided in the Comma Separated Values (CSV) format. Table 4 shows what the columns contain as well as the units of each column. All data can be downloaded from the aurelappius/quad_logs repository on GitHub.

time [s]	x [m]	y [m]	z [m]	v_x [m/s]	v_y [m/s]	v_z [m/s]
⋮	⋮	⋮	⋮	⋮	⋮	⋮

roll [deg]	pitch [deg]	yaw [deg]	ω_{roll} [rad/s]	ω_{pitch} [rad/s]	ω_{yaw} [rad/s]
⋮	⋮	⋮	⋮	⋮	⋮

motor speed$_{1-4}$ $[RPM]$				motor voltage$_{1-4}$ $[V]$				motor current$_{1-4}$ $[A]$			
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 4: Content and units of all columns found in the CSV data.

Appendix C Checklist for Result Recreation

The main idea of this work was the creation of a low-effort but also high-accuracy method for other scientists and engineers to come up with a ground effect model for their quadcopter system. The high-accuracy part was shown in the thesis by conducting thorough experimental validation. The low-effort part is made possible by the following guide that should allow anyone to easily recreate my work. All the code mentioned in the following guide is accessible on GitHub in the aurelappius/quad_control repository.
The following system requirements have to be met to achieve a successful recreation.

system requirements:

- Flight controller that supports MAVLink communication
- Onboard computer capable of compiling and running C++ code with a serial connection to the flight controller.
- Installed dependencies: *MAVSDK*, *Eigen 3*, *yaml-cpp*
- ESC's that support DShot and ESC-Telemetry.

The following high level steps have to be followed to get the ground effect model and the resulting control performance improvement. Always check the code before running it and verify that your quadcopter has enough space to perform the required maneuvers.

checklist:

1. Determine throttle−thrust−rpm relation if not known from a data sheet. The setup described in this thesis in section 3.3 could be used to do so. Enter the information into quad_control.cpp and ground_effect_model.py
2. Run quad_control on the companion computer to fly the *staticDataCollection* mission, which should output log-files to the log directory.
3. Use ground_effect_model.py to generate ground effect model based on the collected flight data.
4. Put model coefficients into compensators.cpp file and activate the compensator in the control loop.

5. Use trajectories.cpp to define and fly custom trajectories.

Feel free to fork the code and change it according to your use-case.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Title of work:

Modeling and Control of Quadcopters in Ground Effect using Physics Informed Machine Learning

Thesis type:

Bachelor's Thesis

Student:

Name:	Aurel X. Appius
E-mail:	appiusa@ethz.ch
Student ID:	19-916-204

Supervisors:

Prof. Dr. Robert K. Katzschmann
Mike Y. Michelis

Declaration of originality

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Signature: Zürich, January 20, 2023